Pettus is concerned with addressing the difficulty of accessing network resources, due to a lack of consistent, globally-accessible directory of network resources that can operate over heterogeneous networks without involving the user in the details and the protocol involved in accessing the separate networks. (col. 4, lines 5-10). Pettus describes a communications directory service located on each node of a network, including a tree structure to which existing director services and other network services can be added. The tree structure has a plurality of nodes each of which includes specific methods that query and browse the associated directory service if such actions are supported by an underlying service. A service object includes the service exchange address and communication link configuration information. A client desiring to access a remote service retrieves the appropriate service object from the communications directory service and uses the service object to set up the communications path. (Pettus, Abstract).

In contrast, claim 1, as amended, now recites "...A method of establishing communication between a first application and a second application, the second application executing on a platform, the method comprising ... forwarding a notify message to the second application, receipt of the notify message by the second application causing the second application to ascertain path data for establishing a path between the first application and the second application, *the notify message including a unique identifier to identify the path, the unique identifier associated with a specific type of information to be transferred on the path*_ ... the first application ascertaining path data for establishing ~~a~~ the path between the first application and the second application; and the first application and second application establishing ~~a~~ the path between the first application and the second application after the path data is ascertained by the first application and the second application..."

Pettus neither describes nor suggests "the notify message including a unique identifier to identify the path, the unique identifier associated with a specific type of information to be transferred on the path..." Rather, Pettus describes only, at column 16, lines 37-47:

> "... a separate data path is set up to send service requests from application program 1100 to the remote service. This separate data path comprises data path 1102, client interface 1126 and the session layer 1123 of the DRPS 1124. In accordance with step 1212, the request information then sent out over physical communication link 1130 to the remote service location."

Thus, Pettus describes using either the address of the session layer or the remote service exchange, but neither describes nor suggests "a unique identifier associated with a specific type of information to be transferred on the path..." Accordingly, for at least this reason, claim 1 is patentably distinct over Pettus, and the rejection should be withdrawn.

Independent claims 13 and 25 have also been amended to include the patentable limitation of "..., the notify message including a unique identifier to identify the path, the unique identifier associated with a specific type of information to be transferred on the path ..." and accordingly the rejection for these independent claims has been overcome as well. Dependent claims 2-12, 14-24 and 26-35 are allowable for at least the reason that they serve to limit patentable parent claims.

Claims 8, 9, 20, 21, 32 and 33

Claims 8, 9, 20, 21, 32 and 33 were rejected under 35 U.S.C. §103(a) as being unpatentable over Pettus in view of Ramanathan (U.S. Patent 6,286,047)

Ramanathan describes a system and method for automatic discovery of network services having two phases of discovery. In a first phase, the services and service elements are detected. In the second phase inter-service dependencies are detected.

Claim 8 recites "... The method as defined by claim 1 wherein the notify message is generated by a monitoring function that monitors the platform, the monitoring function responsively generating the notify message upon detecting that the first application has been added to the platform..." Claims 20 and 32 include a similar limitation.

Applicants note that claims 8, 9, 20, 21, 32 and 33 depend upon independent claims 1, 13 and 25, which are patentable for the reasons stated above. Accordingly, for at least the reasons put forth with regard to their respective parent claims, these claims are patentable as well.

Claims 4, 16 and 28:

Claims 4, 16 and 28 were rejected under 35 U.S.C. §103(a) as being unpatentable over Pettus in view of Aldred (U.S. patent No. 5,539,886).

Aldred describes the concept of application sharing sets. Aldred states: "... Applications are expected to collaborate with other applications, and the mechanism for this collaboration is that they join each other in named application sharing sets. The essence of such an application sharing set is that all set members receive information in the status of all the other members; joining a set is the way in which applications declare those in which they have an interest..."

Claim 4 recites "...the first application forwarding a first ready message to the second application, the second application forwarding a second ready message to the first application... forwarding messages between the first and second application via the path after receipt of each ready message..." Claims 16 and 28 include similar limitations.

The Examiner admits, at page 3 of the office action, that Pettus does not teach forwarding ready messages. The Examiner states "... Aldred teaches (column 31, lines 1-20) ready messages (SHARE_CONFIRMED). It would have been obvious to apply ready messages as taught by Aldred to the invention of Pettus because it would allow the connected applications to notify each other that they are ready to start communications.

Combination neither describes nor suggests claimed invention

As discussed above, Pettus describes an architecture for directory services, in which data paths are set up using a remote service location. Aldred describes a different architecture,

wherein applications can be 'shared'. However, the combination of the two neither describes nor suggests the limitations of the claimed parent, of "..., *the notify message including a unique identifier to identify the path, the unique identifier associated with a specific type of information to be transferred on the path*_ ..." Accordingly, for at least this reason the rejection is overcome and should be withdrawn.

Applicants have made a diligent effort to place the claims in condition for allowance. However, should there remain unresolved issues that require adverse action, it is respectfully requested that the Examiner telephone Lindsay McGuinness, Applicants' Attorney at 978-264-6664 so that such issues may be resolved as expeditiously as possible.

For these reasons, and in view of the above amendments, this application is now considered to be in condition for allowance and such action is earnestly solicited.

Respectfully Submitted,

6/20/2003
Date

Lindsay G. McGuinness, Reg. No. 38,549
Attorney/Agent for Applicant(s)
Steubing McGuinness & Manaras LLP
30 Nagog Park Drive
Acton, MA 01720
(978) 264-6664

Docket No. 120-035
Dd: 06/14/2003

CLAIMS

A1

1. (Currently amended) A method of establishing communication between a first application and a second application, the second application executing on a platform, the method comprising:

forwarding a notify message to the second application, receipt of the notify message by the second application causing the second application to ascertain path data for establishing a path between the first application and the second application, the notify message including a unique identifier to identify the path, the unique identifier associated with a specific type of information to be transferred on the path ;

the first application ascertaining path data for establishing ~~a~~ the path between the first application and the second application; and

the first application and second application establishing ~~a~~ the path between the first application and the second application after the path data is ascertained by the first application and the second application.

2. (Original) The method as defined by claim 1 further comprising:

forwarding a reply message to the first application, the reply message notifying the first application that the second application is executing.

3. (Original) The method as defined by claim 2 wherein the first application ascertains the path data after receipt of the reply message.

4. (Original) The method as defined by claim 1 further comprising:

the first application forwarding a first ready message to the second application;

the second application forwarding a second ready message to the first application;

forwarding messages between the first and second application via the path after receipt of each ready message.

5. (Original) The method as defined by claim 1 wherein the first application and the second application establish a path by ascertaining the path data from a configuration file that includes the path data.

6. (Original) The method as defined by claim 5 wherein the path data is retrieved from the configuration file by the first application and the second application.

7. (Original) The method as defined by claim 5 wherein the path data is retrieved from the configuration file by a path function that forwards a path message to the first application and the second application, the path message including the path data.

8. (Previously Amended) The method as defined by claim 1 wherein each message forwarded between applications includes data identifying the path and channel associated with the message.

9. (Original) The method as defined by claim 8 wherein the first application is considered to have been added to the platform when it is loaded into a volatile memory device on the platform.

10. (Original) The method as defined by claim 1 wherein the second application is considered to be executing after the second application is initialized.

11. (Original) The method as defined by claim 1 wherein an application is considered to be executing after it is initialized on the platform and before it stops running.

12. (Original) The method as defined by claim 1 wherein the path includes a plurality of channels wherein each channel includes an associated handler function, each handler function processing messages in its assigned channel in a uniform manner.

13. (Currently amended) An apparatus for establishing communication between a first application and a second application, the second application executing on a platform, the apparatus comprising:

a first output that forwards a notify message to the second application, receipt of the notify message by the second application causing the second application to ascertain path data for establishing a path between the first application and the second application, the notify message

including a unique identifier to identify the path, the unique identifier associated with a specific type of information to be transferred on the path;

a first controller that controls the first application to ascertain path data for establishing a path between the first application and the second application; and

a second controller that controls the first application and second application to establish a path between the first application and the second application after the path data is ascertained by the first application and the second application.

14. (original) The apparatus as defined by claim 13 further comprising:

a second output that forwards a reply message to the first application, the reply message notifying the first application that the second application is executing.

15. (Original) The apparatus as defined by claim 14 wherein the first application ascertains the path data after receipt of the reply message.

16. (Original) The apparatus as defined by claim 13 further comprising:

a third controller that controls the first application to forward a first ready message to the second application;

a fourth controller that controls the second application to forward a second ready message to the first application, messages being forwarded between the first and second application via the path after receipt of each ready message.

17. (Original) The apparatus as defined by claim 13 wherein the second controller includes a path data ascertainer that ascertains the path data from a configuration file that includes the path data.

18. (Previously Amended) The interface as defined by claim 12 wherein each message forwarded between the applications includes data identifying the path and channel associated with the message.

A1

19. (Original) The apparatus as defined by claim 17 wherein the path data is retrieved from the configuration file by a path function that forwards a path message to the first application and the second application, the path message including the path data.

20. (Original) The apparatus as defined by claim 13 wherein the notify message is generated by a monitoring function that monitors the platform, the monitoring function responsively generating the notify message upon detecting the first application has been added to the platform.

21. (Original) The apparatus as defined by claim 20 wherein the first application is considered to have been added to the platform when it is loaded into a volatile memory device on the platform.

22. (Original) The apparatus as defined by claim 13 wherein the second application is considered to be executing after the second application is initialized.

23. (Original) The apparatus as defined by claim 13 wherein an application is considered to be executing after it is initialized on the platform and before it stops running.

24. (Original) The apparatus as defined by claim 13 wherein the path includes a plurality of channels wherein each channel includes an associated handler function, each handler function processing messages in its assigned channel in a uniform manner.

25. (Currently amended) A computer program product for use on a computer system for establishing communication between a first application and a second application, the second application executing on a platform, the computer program product comprising a computer usable medium having computer readable program code thereon, the computer readable program code including:

program code for forwarding a notify message to the second application, receipt of the notify message by the second application causing the second application to ascertain path data for establishing a path between the first application and the second application, the notify message

including a unique identifier to identify the path, the unique identifier associated with a specific type of information to be transferred on the path;

A1

program code for controlling the first application to ascertain path data for establishing a path between the first application and the second application; and

program code for controlling the first application and second application to establish a path between the first application and the second application after the path data is ascertained by the first application and the second application.

26. (Original) The computer program product as defined by clam 25 further comprising:

program code for forwarding the reply message to the first application, the reply message notifying the first application that the second application is executing.

27. (Original) The computer program product as defined by claim 26 wherein the first application ascertains the path data after receipt of the reply message.

28. (Original) The computer program product as defined by claim 25 further comprising:

program code for controlling the first application to forward a first ready message to the second application;

program code for controlling the second application to forward a second ready message to the first application;

program code for forwarding messages between the first and second application via the path after receipt of each ready message.

29. (Original) The computer program product as defined by claim 25 wherein the program code for controlling the first application and the second application comprises program code for ascertaining the path data from a configuration file that includes the path data.

30. (Previously Amended) The computer program product as defined by claim 23 wherein each message forwarded between applications includes data identifying the first path and channel associated with the message.

31. (Original) The computer program product as defined by claim 29 wherein the path data is retrieved from the configuration file by a path function that forwards a path message to the first application and the second application, the path message including the path data.

32. (Original) The computer program product as defined by claim 25 wherein the notify message is generated by a monitoring function that monitors the platform, the monitoring function responsively generating the notify message upon detecting that the first application has been added to the platform.

33. (Original) The computer program product as defined by claim 32 wherein the first application is considered to have been added to the platform when it is loaded into a volatile memory device on the platform.

34. (Original) The computer program product as defined by claim 25 wherein the second application is considered to be executing after the second application is initialized.

35. (Original) The computer program product as defined by claim 25 wherein an application is considered to be executing after it is initialized on the platform and before it stops running.

36. (Original) The computer program as defined by claim 25 wherein the path includes a plurality of channels wherein each channel includes an associated handler function, each handler function processing messages in its assigned channel in a uniform manner.